Protokolle zur Anbindung kundenlokaler Infrastruktur an eine Cloud-Umgebung

Sebastian Steinbuß, Gregor Kasmann

Fraunhofer-Institut für Software- und Systemtechnik
Emil-Figge-Str. 91
44227 Dortmund
Sebastian.Steinbuss@isst.fraunhofer.de
Gregor.Kasmann@cs.tu-dortmund.de

Abstract: Im Logistikumfeld ist der Einsatz von Cloud-Anwendungen problematisch. Häufig werden physische Ressourcen wie Drucker, Scanner, Förder- oder Lagertechnik durch eine Anwendung angesprochen. Beim Einsatz einer Cloud-Anwendung kommunizieren Anwendung und Gerät über das Internet miteinander. In dieser Arbeit wird für die Kommunikation der Partner das REST-Architekturmodell vorgeschlagen und mit einem SOAP-basierten Modell verglichen. Der REST-Ansatz wird an einem Beispiel verdeutlicht.

1 Peripherieanbindung an Cloud-Systeme

Die Nutzung von Anwendungen aus der Cloud ist insbesondere für kleine und mittelständische Unternehmen (KMU) interessant. Diese können häufig keine oder nur eine kleine IT-Abteilung unterhalten. Zusätzlich ist das Vorhalten von IT-Infrastruktur mit Investitionskosten und laufenden Kosten verbunden. Durch das Cloud Computing werden die benötigten IT-Ressourcen in einer einfachen Weise mit nutzungsorientierten Abrechnungsmodellen zur Verfügung gestellt [MU09]. Dementsprechend ist Cloud Computing nicht nur eine Technologie, sondern auch ein neues Geschäftsmodell nach dem der Computer als Werkzeug (Utility Computing) eingesetzt werden kann [AR09, RE10]. Für das Cloud Computing existiert keine einheitliche Definition. In dieser Arbeit wird die Definition vom National Institute of Standards and Technology (NIST) zugrunde gelegt, die die allgemein akzeptierten Charakteristika des Cloud Computing beschreibt. Cloud Computing ist ein Modell um komfortablen, bedarfsgerechten Zugang zu einem Pool von konfigurierbaren Rechnerressourcen (Netzwerke, Server, Speicherkapazität, Anwendungen und Services) über das Netzwerk zu erhalten, wobei die Ressourcen kurzfristig zugewiesen und freigegeben werden können, so dass ein minimaler Aufwand und minimale Interaktion mit dem Dienstanbieter nötig ist. Das Cloud-Modell ist durch fünf Charakteristika beschrieben, umfasst drei Dienstmodelle und vier Zugangsmodelle [ME09].

Charakteristisch für Cloud-Angebote sind die bedarfsorientierte Anforderung von Ressourcen durch den Kunden, ein breitbandiger Zugang zu dem Angebot, der Zusammenschluss von virtuellen Ressourcen um diese multi-mandantenfähig zugänglich zu machen, schnelle Elastizität der Ressourcen und die nutzungsorientierte Abrechnung. Die Dienstmodelle umfassen Infrastructure as a Service (IaaS), Plattform as a Service (PaaS) und Software as a Service (SaaS). Eine Cloud-Umgebung kann nur unternehmensintern verfügbar sein (Private Cloud), ein Zusammenschluss von Unternehmensressourcen sein (Community Cloud) oder von einem anderen Unternehmen für verschiedene Nutzergruppen bereitgestellt werden (Public Cloud). Die Kombination von privaten und öffentlichen Clouds wird als hybride Cloud bezeichnet [LE09, MU09, ME09]. Im Fokus dieser Arbeit liegen öffentliche und hybride Clouds.

Der Übergang zu Cloud-oder SaaS-Angeboten ist für die Anwender in vielen Unternehmen einfach, da es für den Nutzer transparent ist, ob ein System von der eigenen IT-Abteilung oder bei einem externen Provider betrieben wird [HA08]. Im Logistikumfeld ist der Übergang zu Cloud-Lösungen schwieriger. Die Software-Systeme interagieren häufig direkt mit der kundenlokalen Infrastruktur, wie Handscannern oder Druckern [TE08]. Die Kommunikation zwischen der Anwendung und der kundenlokalen Infrastruktur muss beim Einsatz einer Public Cloud-Lösung über das Internet erfolgen. Von dieser Änderung sind die Kommunikationsprotokolle direkt betroffen, da Anwendung und Gerät über das Internet, statt wie heute über das Intranet, miteinander kommunizieren. Die netzübergreifende Kommunikation verlangt Überlegungen zu Routing, Namensauflösung und Sicherheit, die bei Intranet-Lösungen nicht nötig sind.

Im Folgenden werden zunächst standardisierte Protokolle zur Anbindung kundenlokaler Infrastruktur vorgestellt und wie REST-Webservices alternativ eingesetzt werden können. Der vorgestellte Ansatz wird an einem Einsatzszenario beispielhaft dargestellt.

2 Protokolle zur Anbindung kundenlokaler Infrastruktur

Bei der Anbindung kundenlokaler Infrastruktur an eine Cloud-Umgebung müssen verschiedene Geräteklassen unterschieden werden. Geräte können als Aktoren auftreten, die nur Daten empfangen, als Sensor, die Daten an die Anwendung senden oder können sowohl Aktor als auch Sensor in sich vereinen und bidirektional kommunizieren [TE08]. Üblicherweise werden in der Logistik Geräte wie Drucker, Barcodescanner oder Waagen eingesetzt, die hauptsächlich unidirektional kommunizieren. Das eingesetzte Kommunikationsprotokoll muss die Nutzdaten über das Internet übertragen können [LE09].

Es existiert eine Reihe von Standardprotokollen zur Kommunikation mit IT-Diensten, z.B. Verzeichnisdiensten, über das Internet, allerdings wenige Protokolle, die die Kommunikation mit physischen Geräten über das Internet unterstützen. Das Anwendungsprotokoll LDAP (Lightweight Directory Access Protocol) erlaubt sowohl die Abfrage eines Verzeichnisdienstes über das Internet, als auch die Modifikation von Einträgen. Im Gegensatz zum X500 Standard, von dem LDAP abgeleitet wurde, basiert LDAP auf dem Internetprotokoll TCP/IP und ist uneingeschränkt über das Internet

verfügbar sowie durch Mechanismen zur Authentifizierung und Autorisation absicherbar [SE06]. Durch schnelle Mechanismen zum Verbindungsaufbau und –abbau ist es für den Einsatz als Autorisierungs- und Authentifizierungskomponente in einem Identity and Access Management System optimiert, sowie für die Abfrage von Nutzer- oder allgemeinen Ressourceninformationen.

Das Anwendungsprotokoll IPP (Internet Printing Protocol) wurde zur Anbindung von Druckdiensten über das Internet entwickelt. Es versendet seine Befehle mittels HTTP 1.1 ohne dafür neue HTTP-Methoden oder Header zu benötigen. Stattdessen werden Befehle und Daten binär kodiert in HTTP-POST-Anfragen mit dem Content-Type "application/ipp" übertragen. IPP wird sowohl von Drucker- als auch von Betriebssystemherstellern breit unterstützt. [HE03].

Das Funktionsprinzip von IPP lässt sich auf allgemeine Ressourcen erweitern, da der HTTP-Standard selbst die wichtigsten Operationen zur Verfügung stellt. Der REST-Architekturstil [FI00] beschreibt, wie Web-Standards, wie HTTP, allgemein eingesetzt werden sollten. REST nutzt die Semantik des HTTP-Protokolls. Es nutzt die durch das HTTP-Protokoll definierten Methoden GET, PUT, POST, DELETE, HEAD und OPTIONS. Mit diesen Methoden müssen alle Anwendungsfälle generisch abgedeckt werden. Die Semantik der HTTP-Methoden bleibt gemäß dem REST-Architekturansatz erhalten. Die Methode GET fragt die Repräsentation einer Ressource frei von Seiteneffekten ab. Der Aufruf der GET-Methode verändert die Ressource nicht und kann daher beliebig oft mit dem gleichen Ergebnis wiederholt werden. Um einer Ressource etwas hinzuzufügen, wird die Methode POST genutzt. Der POST-Aufruf verändert eine Ressource und ist damit nicht frei von Seiteneffekten, erzeugt aber bei mehrfacher Ausführung mit der gleichen Eingabe die gleiche Ausgabe. Die Methode PUT erzeugt eine neue Ressource oder ändert diese, DELETE entfernt eine Ressource. Beide Methoden sind dementsprechend nicht frei von Seiteneffekten. Mit diesen Methoden lassen sich die elementaren CRUD (Create, Read, Update, Delete) Operationen abdecken. Die bei einem Methodenaufruf übertragenen Nutzdaten können von einem beliebigen Format sein. Für die meisten Datenformate existiert ein Content-Type, der das Datenformat beschreibt. Zur strukturierten Datenübertragung eignet sich das XML-Format, allerdings ist die Wahl des richtigen Datenformates vom Anwendungsfall abhängig.

Die wenigsten Geräte besitzen eine Implementierung einer REST-Schnittstelle, so dass zumeist ein Webserver, z. B. Apache Tomcat oder Microsoft IIS, die REST-Schnittstelle als Proxy bereitstellt, Daten an das Gerät weiterleitet und in ein gerätespezifisches Format umwandelt. Die Anbindung einer lokalen Kunderessource über REST erfordert keinen großen Entwicklungsaufwand, denn im Kern unterscheidet sich ein REST-Server nicht von anderen Anwendungen. Der zentrale Unterschied liegt in der Ausprägung der Schnittstelle, die die Ressource kapselt. Der Client in der Cloud nutzt das HTTP-Protokoll um Geschäftsobjekte zu übertragen. Diese können als XML-Dokument kodiert werden. Die auf der Ressource benötigten Methoden müssen auf die HTTP-Methoden abgebildet werden und es muss ein geeigneter Server implementiert werden.

Tritt eine Cloud-Anwendung als REST-Server auf, so sind hier grundlegende Designentscheidungen zu treffen, die in der Natur einer Cloud-Anwendung begründet sind und ausgesprochen gut mit dem REST-Architekturstil harmonieren. Die Methoden einer Cloud-Anwendung sollten atomar, zustandslos, idempotent und auf massive Parallelverarbeitung ausgelegt sein [SM09]. Die Forderung nach Zustandslosigkeit wird ebenfalls beim REST-Ansatz gefordert, genauso wie die Forderung nach Idempotenz. Die HTTP-Methoden können mit der gleichen Eingabe mehrfach auf einer Ressource durchgeführt werden und es wird die gleiche Ausgabe erzeugt. Durch die Bereitstellung von atomaren Methoden wird das Programm unabhängig von Ausführungsreihenfolge der Methoden. Auch dies unterstütz den REST-Ansatz, bei dem Client und Server nur ihren eigenen Zustand kennen und der Client für die korrekte Aufrufreihenfolge verantwortlich ist, die von der Aufgabe des Clients abhängt und nicht durch den Server vorgegeben werden sollte. Die letzte Forderung nach Programmcode, der für die massive Paralleleverarbeitung optimiert ist, ist ausschließlich der Natur von Cloud-Anwendungen geschuldet, die in einer massiv parallelisierten Umgebung ausgeführt wird.

Der Einsatz von SOAP-basierten Webservices ist eine Alternative zu REST [ZM05]. Bei SOAP wird HTTP oder SMTP als Transportprotokoll genutzt. Die Nachrichten werden im XML-Format ausgetauscht, wobei aber auch Dokumente als Anhang mitgeschickt werden können. Allerdings werden die SOAP-Nachrichten immer an einen zentralen Dispatcher geschickt, der den Empfänger aus der SOAP-Nachricht extrahiert. Der grundsätzliche Unterschied zwischen REST und SOAP ist also die Art der Adressierung. Mit REST können Ressourcen direkt adressiert werden, während bei SOAP ein Service adressiert wird, der die Nachrichten an die Ressource weiterleitet. SOAP gibt dem Entwickler allerdings den Freiraum ein vollständiges eigenes Kommunikationsprotokoll zu entwickeln, in dem Anfrage und Antwort fest vorgegeben sind und in einem Vertrag, z. B. durch WSDL, bekannt gegeben und zugesichert werden. [GU07].

Durch die nachrichtenorientierte Architektur ist die bidirektionale Kommunikation zwischen einer kundenlokalen Infrastruktur-Ressource und einer Cloud-Umgebung möglich. Die Kommunikation ist allerdings asynchron und durch das Routing über das Internet und Datentransformation in das XML-Format bzw. Serialisierung mit Latenzen behaftet. Dies sollte bei der Entwicklung eines Systems, dass eine Cloud-Umgebung bidirektional an eine kundenlokale Infrastruktur anbindet, bereits beim Systemdesign beachten werden. Für Anwendungsfälle, die eine synchrone Kommunikation erfordern, müssen separate Überlegungen angestellt werden. Grundsätzlich ist zu prüfen, ob die Ausgliederung einer solchen Funktionalität in die Cloud überhaupt sinnvoll ist. So kann es attraktiv sein ein Lagerverwaltungssystem in der Cloud zu betreiben, den Materialflussrechner mit Anbindung an ein Regalbediengerät aber weiterhin lokal zu betreiben um die Latenzen gering zu halten. Die Kommunikation zwischen Materialflussrechner und Lagerverwaltung ist nicht zeitkritisch und kann asynchron erfolgen, im Gegensatz zur Kommunikation zwischen Materialflussrechner und Regalbediengerät, die sowohl zeitkritisch ist als synchron erfolgen sollte.

3 Einsatzszenarien

Der REST-Ansatz lässt sich an dem Beispiel einer Lagerverwaltung, die ein Regalbediengerät über einen Materialflussrechner steuert, gut verdeutlichen. Jedes Fach in einem Regal lässt sich eindeutig identifizieren und als Ressource ansprechen. Dabei kann die Ressource Fach eine untergeordnete Ressource zu der Ressource Regal sein. Durch die GET-Methode werden Informationen zu der Ressource angefordert. In der Antwort können die Abmessungen des Fachs enthalten sein und ein Verweis auf den aktuellen Inhalt. Ein Client kann dem Verweis auf den Inhalt folgen und weitere Informationen aus einer Datenbank erhalten. Durch den Aufruf der GET-Methode wird der Inhalt aber nicht aus dem Fach entnommen. Das Regalbediengerät kann durch einen POST-Aufruf ein Paket in ein Regalfach einlegen. Das Paket lässt sich durch einen Verweis spezifizieren, ebenso der aktuelle Lagerort. Abhängig vom Anwendungsfall und Systemdesign kann noch der Aufruf der DELETE-Methode auf dem vorherigen Lagerort nötig sein. Wird ein Paket am Wareneingang angenommen, so wird es durch die Methode PUT der Repräsentation des Wareneingangs, beispielsweise einer Datenbank, hinzugefügt. Dabei wird die neue Ressource Paket erzeugt und der Ressource Wareneingang hinzugefügt. Solche Cloud-Anwendungen für die Unterstützung von Logistikprozessen werden beispielsweise in dem Fraunhofer Innovationscluster „Logistics Mall – Cloud Computing für die Logistik“ (www.logistics-mall.de) und dem BMBF geförderten Spitzencluster-Wettbewerb Gewinner „EffizienzCluster LogistikRuhr“ (www.effizienzcluster.de) unter dem Leitthema „Logistics as a Service“ erforscht und entwickelt [RT10, TE08a].

4 Fazit

Die Anbindung kundenlokaler Infrastruktur an eine öffentliche Cloud-Umgebung erfordert geräte- und umgebungsspezifische Entwicklungsleistung. Für wenige Infrastrukturkomponenten existieren Protokolle, die von beiden Kommunikationspartnern unterstützt werden. Der REST-Architekturansatz ist bei der Eigenentwicklung von Kommunikationsprotokollen dem SOAP-Ansatz vorzuziehen. REST setzt Internet-Technologien ein und harmonisiert gut mit den softwaretechnischen Grundsätzen im Cloud Computing. Die durch eine REST-basierte Ressource angebotenen Services lassen sich gut zu komplexen Prozessen orchestrieren. Dies resultiert aus den Eigenschaften des REST-Ansatzes, insbesondere Atomarität und Zustandslosigkeit. Der Transport über das HTTP-Protokoll erleichtert den Einsatz zusätzlich, da dies wenige Eingriffe in die Sicherheitsarchitektur der beteiligten Unternehmen zur Folge hat und und von Standardnetzwerktechnik unterstützt wird. Die Public Cloud Provider Google und Amazon unterstützten die sichere Verbindung von Cloud und Unternehmens-netzwerk bereits durch die Amazon Virtual Private Cloud und den Google Secure Data Connector.

In der Cloud verschwimmt der Ressourcenbegriff, da die Ressourcen in der Cloud zwar elastisch sind, aber doch an eine physische Hardwareumgebung gebunden sind. Beim Senden von Daten an eine Cloud-Anwendung wird die Schnittstelle adressiert. Diese leitet die Anfrage an die Ressource weiter, die für die Verarbeitung verantwortlich ist. Dies erfordert eine entsprechende Konfiguration Cloud-internen IP-Netze, insbesondere bei der Namensauflösung und der Adressübersetzung (Network Adress Translation - NAT) Der Provider ist an dieser Stelle in der Verantwortung genügend Ressourcen bereit zu stellen um Dienstgütevereinbarungen (Service Level Agreement - SLA) [VE04] einzuhalten und muss an dieser Stelle abwägen, ob es möglich ist die Verarbeitung abzulehnen, zu pausieren oder eine andere Anfrage abzubrechen.

Literaturverzeichnis

[AR09] Armbrust, M. et al.: Above the Clouds: A Berkeley View of Cloud Computing, Technical Report No. UCB/EECS-2009-28, University of California at Berkley, USA, 2009.

[FI00] Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. PhD-Thesis, University of California, Irvine, 2000

[GU07] Gudgin, M. et al.: SOAP Version 1.2 Part 1: Messaging Framework (Second Edition), W3C Recommendation, 2007

[HA08] Hayes, B.: Cloud Computing. Communications of the ACM, 51(7), 2008

[HE03] Herriot, R.; McDonald, I: Internet Printing Protocol/1.1: IPP URL Scheme. RFC3510, 2003

[LE09] Lenk, A. et al.: What's inside the Cloud? An architectural map of the Cloud landscape. CLOUD '09: Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges (IEEE Computer Society). 2009 S. 23--31

[ME09] Mell, P.; Grance, T.: The NIST Definition of Cloud Computing, Working Paper National Institute of Standards and Technology, 2009, http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc .

[MU09] Münzl, G. et.al.: Cloud Computing – Evolution in der Technik, Revolution im Business BITKOM Leitfaden. Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V. (Hrsg.)-Berlin, 2009

[RE10] Rehof, J; Gottschick, J: Anwendungen aus der Steckdose, Der Neue Kämmerer, 01/2010, S. 3

[RT10] Rehof, J; Ten Hompel, M: IT aus der Wolke, Log.Kompass, 03/2010, S. 28 -29

[SE06] Sermersheim, J (Hrsg): Lightweight Directory Access Protocol (LDAP): The Protocol. RFC4511, 2006

[SM09] Semones, G.: Building cloud-ready, multicore-friendly applications, JavaWorld.com, 2009

[TE08] Ten Hompel, M; Schmidt, T: Warehouse Management - Organisation und Steuerung von Lager- und Kommissioniersystemen. Springer-Verlag Berlin Heidelberg, 2008

[TE08a] Ten Hompel, M. :Logistics by Design-ein Ansatz für zukunftsfähige Intralogistik In: Hebezeuge Fördermittel 5/2008, S. 236-240

[VE04] Verma, D.C. : Service Level Agreements on IP Networks. Proceedings of the IEEE, 92(9) , 2004

[ZM05] Zur Muehlen, Michael; Nickerson, J. V.; Swenson, K. D.: Developing web services choreography standards--the case of REST vs. SOAP. Decision Support Systems 40(1), S. 9-29